PyBioC: a python implementation of the BioC core

Marques, Hernani ; Rinaldi, Fabio

PyBioC: a python implementation of the BioC core

Hernani Marques, Fabio Rinaldi

Motivation

BioC[1] (1) is a recently proposed framework which aims at providing a simple and yet powerful approach for the integration of text mining tools, based on a combination of an XML-based data interchange format, and the implementation of a library that allows memory-based handling of documents at all levels of processing. Implementations of the BioC framework have been provided in Java and C++.

OntoGene is a specialized text mining system for Named Entity Recognition (NER) and relationship extraction, capable of dealing with a variety of entities, such as chemicals, diseases, drugs, genes or proteins. The effectiveness of the system has been tested in several text mining evaluations, in which the OntoGene team has participated with succcess. Best results have been achieved in the detection of experimental methods (BioCreative 2006), in the detection of interactions between proteins (BioCreative 2009), in large-scale detection of biomedical entities for some entity categories (CALBC 2010). In the CTD triage task of BioCreative 2012 the OntoGene system provided best overall results.

The OntoGene system (3,4) is based on a relatively heterogeneous pipeline, composed of tools implemented in perl, xslt, python and prolog. The different modules of the pipeline exchange text annotations as I/O files in a common XML format, which has several similarities with the proposed BioC XML format. The original idea of the system was to allow memory-based sharing of annotations, but this was never implemented due to the heterogeneity of the components. Recently most of the modules (but not yet all of them) have been reimplemented in python, which would allow memory-based processing. We plan to switch gradually to a BioC compatibile internal format and for this purpose we have decided to provide an independent implementation of the BioC core in python.

Additionally, as part of our participation in task 3, we have also implemented a RESTful (2) web service which allows submission of input documents in BioC format and delivers entity annotations in the same format.

[1] http://www.ncbi.nlm.nih.gov/CBBresearch/Dogan/BioC/

PyBioC

In BioCreative IV Track-1 participants were asked to contribute to the BioC (BioCreative) community in the area of interoperability. The Ontogene team based in Zurich was confronted with the fact that no native BioCreative library for use with the Python programming language was available until now. To our knowledge no other team or initiative aimed at changing this, such that we took up this opportunity to create a Python implementation of the BioC library.

The PyBioC library recreates the functionality of the already available libraries in C++ or Java. However, we adhere to Python conventions were suitable, for example refraining from implementing getter or setter methods for internal variables of the classes provided in PyBioC.

Basically the library consists of a set of classes representing the minimalistic data model proposed by the BioC community. Two specific classes (BioCReader and BioCWriter) are available to read in data provided in (valid) XML format and to write from PyBioC objects to valid BioC XML format. Validity is ensured by following the BioC DTD publicly available.

The library is being developed as free software and is available on a public github repository (https://github.com/2mh/PyBioC), where example programs can be found, specifically to read in and write to BioC XML format or to tokenize and stem a given BioC XML input file using the Natural Language Toolkit (NLTK) library. As an example of an application, we also provide the integration of a standard word stemmer in PyBioC:

https://github.com/2mh/PyBioC/blob/master/src/stemmer.py

Conclusion

PyBioC enables the biomedical text mining community to deal with BioC XML documents using a native implementation of the BioC library in the Python programming language. PyBioC is available as open-source under the Simplified BSD License. We welcome further contributions and additions to this work.

Our contribution to task 1 (a python implementation of the BioC core) is available at:
https://github.com/2mh/PyBioC/tree/master/src/biocß

Acknowledgements

The OntoGene group is partially supported by the Swiss National Science Foundation (grants 100014−118396/1 and 105315−130558/1).

References

1. Donald C. Comeau, Rezarta Islamaj Doğan, Paolo Ciccarese, Kevin Bretonnel Cohen, Martin Krallinger, Florian Leitner, Zhiyong Lu, Yifang Peng, Fabio Rinaldi, Manabu Torii, Alfonso Valencia, Karin Verspoor, Thomas C. Wiegers, Cathy H. Wu, W. John Wilbur (2013). BIoC: a minimalist approach to interoperability for biomedical text processing, The Journal of Biological Databases and Curation (2013), *bat064, doi:10.1093/database/bat064, published online.*
2. Richardson, Leonard; Ruby, Sam (2007), *RESTful Web Services*, O'Reilly, ISBN 978-0-596-52926-0
3. Fabio Rinaldi, Thomas Kappeler, Kaarel Kaljurand, Gerold Schneider, Manfred Klenner, Simon Clematide, Michael Hess, Jean-Marc von Allmen, Pierre Parisot, Martin Romacker, Therese Vachon (2008). OntoGene in BioCreative II. Genome Biology, 2008, 9:S13, PMC2559984
4. Fabio Rinaldi, Gerold Schneider, Kaarel Kaljurand, Simon Clematide, Thérèse Vachon, Martin Romacker, "OntoGene in BioCreative II.5," IEEE/ACM Transactions on Computational Biology and Bioinformatics, 7(3), pp. 472-480, 2010. http://doi.ieeecomputersociety.org/10.1109/TCBB.2010.50